REMARKS

In the Office Action dated September 15, 2004, claims 1-4, 6-8, 10-19, 21-31, and 33-36 were rejected under 35 U.S.C. § 103 over U.S. Patent No. 6,128,640 (Kleinman) in view of U.S. Patent No. 5,706,515 (Connelly); claims 5 and 32 were rejected under § 103 over Kleinman in view of Connelly and U.S. Patent No. 5,598,562 (Cutler); claims 9 and 20 were rejected under § 103 over Kleinman in view of Connelly and U.S. Patent No. 6,546,443 (Kakivaya); and claims 37-38 were rejected under § 103 over Kleinman.

As amended, claim 1 is not rendered obvious over the asserted combination of Kleinman and Connelly. Claim 1 now recites a system having a Unix operating system, a plurality of execution entities including a first execution entity, and an event control module adapted to create event objects representing respective events each having a state. The first execution entity waits on plural events, with a data structure associated with the first execution entity containing information of the plural events that the first execution entity is waiting on. The data structure further contains an indicator *settable to one of plural values to specify respect plural logical relationships* between the plural events. Moreover, the system includes a controller to awaken the first execution entity by signaling the first execution entity in response to one or more event state changes of the states of the plural event according to *the logical relationship specified by the indicator.*

In the obviousness rejection, the Office Action referred to the Alert object and the Thread_Exit_Event object as teaching event objects. 9/15/2004 Office Action at 2. Kleinman describes two types of Alert objects: a container Alert object, and a contained Alert object. The container Alert object has a non-null collection of Alert objects, each of which corresponds to an event. Kleinman, 4:29-30. "When a thread blocks on a single Alert object of the container type, the thread effectively waits on the *logical inclusive OR* of the events represented by all of the Alert objects in the container. The thread unblocks when *any* of these events occur." Kleinman, 4:31-35 (emphasis added). Thus, the container of the Alert object in Kleinman allows a thread to wait on the logical OR of the events represented by the Alert objects in the container. Kleinman does not teach that a container includes, or is associated with, an indicator that is settable to one of plural values to specify respective plural logical relationships. Connelly also fails to disclose or suggest this missing element. Therefore, a *prima facie* case of obviousness cannot be

established with respect to claim 1 over the asserted combination of Kleinman and Connelly. Withdrawal of the § 103 rejection is therefore respectfully requested.

Amended independent claim 28 is similarly allowable over Kleinman and Connelly. Therefore, withdrawal of the § 103 rejection against claim 28 is also respectfully requested.

Dependent claims 2-9, 12-20, 29-36, 39, 40, 43, and 44 are allowable for at least the same reasons as corresponding independent claims 1 and 28. Moreover, dependent claims 39 and 43 further define that the indicator is settable to a first value to specify a logical AND relationship between the plural events, a feature not suggested by either Kleinman or Connelly.

Amended independent claim 21 is also allowable over the asserted combination of Kleinman and Connelly. Claim 21 recites the provision of a queue containing entries associated with a first event object, each entry associated with a corresponding execution entity, where the plural entries of the queue enable plural execution entities to wait on the first event object. Moreover, claim 21 recites that a type variable is selectively set to one of a first value and a second value, with the first value indicating that the first event object is of an auto-reset type, and the second value indicating that the second event object is of a manual reset type. In response to the state of the first event object indicating the corresponding event has been signaled, the system automatically clears the state of the first event object to un-signaled state and awakens only one of the plural execution entities waiting on the first execution object in response to the type variable being set to the first value. However, if the type variable is set to the second value, then the state of the first event object is not cleared until manually cleared, and all threads waiting on the first event object are awakened.

This type variable is not disclosed or suggested by either Kleinman or Connelly. However, the Office Action has indicated that Kakivaya teaches such a type indication (obviousness rejection of dependent claims 9 and 20).

Applicant respectfully submits that there existed no motivation to combine the teachings of Kleinman, Connelly, and Kakivaya. There are at least two reasons not to combine the teachings Kleinman and Kakivaya. First, Kleinman relates to the Unix operating system, whereas Kakivaya relates to a Windows operating system. Second, Kleinman uses a notify_all function to unblock (or awaken) all threads waiting for a particular event object. Kleinman, 5:53-54, 8:37-41. Kleinman makes no suggestion whatsoever of any need to selectively awaken

just one thread, or awaken all threads, based on the type of event object. Except for impermissible hindsight reconstruction based on the teachings of the present invention, no suggestion existed in the prior art references to modify the Kleinman system to incorporate the teachings of Kakivaya.

Moreover, due to the fact that Kleinman and Kakivaya relate to different types of operating systems, it is not readily apparent that a person of ordinary skill in the art would have recognized the techniques applicable to one type of operating system can be employed in a different operating system environment. The Office Action has failed to explain how a person of ordinary skill in the art would have recognized that the Kakivaya Windows event object mechanism can be applied to the Kleinman Unix operating system event object mechanism.

For the foregoing reasons, it is respectfully submitted that a *prima facie* case of obviousness cannot be established with respect to claim 21 over the asserted combination of Kleinman, Connelly, and Kakivaya.

Dependent claims 22-26, 41, and 42, which depend from claim 21, are allowable for at least the same reasons. Moreover, claim 41 recites the provision of a data structure containing information of plural events waited upon by the first execution entity, where the data structure further contains an indicator settable to one of plural values to specify respective *plural logical relationships* between plural events waited upon by the first execution entity. As explained above with respect to claim 1, neither Kleinman nor Connelly teaches or suggests this particular feature of claim 41.

In view of the foregoing, allowance of all claims is respectfully requested. The Commissioner is authorized to charge any additional fees and/or credit any overpayment to Deposit Account No. 50-1673 (9491).

Respectfully submitted,

Date: 12-15-04

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, TX 77024
Telephone: (713) 468-8880
Facsimile: (713) 468-8883